Can Large Language Models Learn Independent Causal Mechanisms?

Gaël Gendron, Bao Trung Nguyen, Alex Yuxuan Peng, Michael Witbrock, Gillian Dobbie

NAOInstitute, University of Auckland
Correspondence: gael.gendron@auckland.ac.nz

Abstract

Despite impressive performance on language modelling and complex reasoning tasks, Large Language Models (LLMs) fall short on the same tasks in uncommon settings or with distribution shifts, exhibiting a lack of generalisation ability. By contrast, systems such as causal models, that learn abstract variables and causal relationships, can demonstrate increased robustness against changes in the distribution. One reason for this success is the existence and use of Independent Causal Mechanisms (ICMs) representing high-level concepts that only sparsely interact. In this work, we apply two concepts from causality to learn ICMs within LLMs. We develop a new LLM architecture composed of multiple sparsely interacting language modelling modules. We show that such causal constraints can improve out-of-distribution performance on abstract and causal reasoning tasks. We also investigate the level of independence and domain specialisation and show that LLMs rely on pre-trained partially domain-invariant mechanisms resilient to fine-tuning.

1 Introduction

The latest generation of Large Language Models (LLMs) with over several billion parameters has demonstrated impressive performance on an extensive range of in-context language and reasoning tasks (Bubeck et al., 2023; Brown et al., 2020; Wei et al., 2022b,a; Touvron et al., 2023a) and an even greater range when fine-tuned for a specific task (Touvron et al., 2023b; Hu et al., 2022). However, these observations do not hold for tasks that fall outside the training data distribution, sometimes even when the task is only slightly perturbed. In particular, standard LLMs perform poorly on complex reasoning tasks, such as abstract, causal, or logical reasoning (Wu et al., 2023; Gendron et al., 2023a; Zecevic et al., 2023; Jin et al., 2023; Liu et al., 2023; Bao et al., 2023). Gendron et al. (2023a); Jin et al. (2023); Wu et al. (2023) showed that fine-tuning LLMs can increase their in-distribution performance, but the improvement does not transfer to different distributions, highlighting that LLMs do not generalise as we might expect a person to when applied to domains requiring complex reasoning. Several hypotheses have been proposed to explain this flaw, such as the lack of abstract or symbolic representations within the latent space of LLMs (Wu et al., 2023; Gendron et al., 2023a; Goyal and Bengio, 2020). These claims are supported by the brittleness that LLMs can exhibit; when changing the wording of a question, the performance of an LLM can vary drastically (Wei et al., 2022b; Jin et al., 2023). This observation hints that LLMs may rely on domain-specific information or spurious correlations in the training data that do not generalise to other distributions.

Causal models rely on the concept that causal mechanisms invariant under changes in environment exist. The *Independent Causal Mechanisms* principle further states that *"the causal generative process of a system's variables is composed of autonomous modules that do not inform or influence each other. "* (Peters et al., 2017; Schölkopf et al., 2021). These principles are applied in diverse ways in the field of causality, either in the structure of the model, which may be built in a modular fashion to respect causal relationships, as in Structural Causal Models (Pearl, 2009), or in the distribution of the data, which may be rendered independent and identically distributed (i.i.d) from an unbalanced distribution by division into subgroups (Austin, 2011; Gendron et al., 2023c). Integrating these methods into the architecture of a Large Language Model could increase its robustness and out-of-distribution (o.o.d) generalisation.

We investigate this idea in this work: we aim to better understand how LLMs reason in and out-of-distribution and whether they can behave as models of Independent Causal Mechanisms un-

Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing, pages 6678–6701
November 12-16, 2024

der certain constraints and with fine-tuning. To this end, we propose an LLM architecture integrating the concept of mechanisms as independent, self-contained LLM modules. This model is summarised in Figure 1. We aim to answer the following questions: (i) Can LLMs be used as self-routers for specialised mechanisms, and does it improve their performance? (ii) Can LLMs capture domain-invariant abstractions with information-based regularisation? (iii) How useful is domain-specific knowledge on reasoning tasks? and (iv) Can our proposed architecture approximate Independent Causal Mechanisms? Our contributions can be summarised as follows:

- We propose a modular LLM architecture yielding modularity and abstraction in LLMs using routing and regularisation mechanisms.
- We investigate the ability of LLMs to behave as Independent Causal Mechanisms on reasoning tasks and show that it can lead to improved performance and o.o.d generalisation.
- We show that LLMs approximate independent mechanism up to an extent but always partially rely on pre-trained domain-invariant mechanisms for reasoning tasks.

Our code is available at `https://github.com/Strong-AI-Lab/modular-lm`.

2 Related Work

LLM Mixtures-of-Experts Modular architectures divide the computations of a network into sub-networks. The Switch Transformer (Fedus et al., 2022b) separates the feed-forward layers of the transformer model (Vaswani et al., 2017) into multiple *expert* modules. This strategy allows training larger models at a lower cost, but the expert modules are not guaranteed to specialise in specific domains. Multiple sparse architectures have followed but mainly focus on optimising the training of LLMs for reduced resources and not inducing domain specialisation (Fedus et al., 2022a). One exception is the work of Gururangan et al. (2022), which conditions the activation of an expert module on the input domain. However, only the feed-forward layers are used as experts and the domains are assumed to be known during training. Clark et al. (2022) investigate the performance of various routing strategies for LLMs and show that the gain from using specialised modules is high for small models but decreases as the model size increases. Introduced recently, Mixtral-of-Experts is a modular LLM using the same routing principle as the Switch Transformer. It outperforms dense LLMs of similar size on reading comprehension, commonsense knowledge and reasoning tasks (Jiang et al., 2024). However, the authors observe that the routing process does not lead to domain-specialised modules. The assignment of experts is not based on domain information. Our work differs in that it is not directed at optimising LLM training but at inducing functional modularity and studying its effects on generalisation for reasoning tasks.

Modular Neural Networks Other classes of modular neural models are designed to learn specialised sub-networks for specific domains. Recurrent Independent Mechanisms (Goyal et al., 2021) attempt to learn models of independent mechanisms with an LSTM architecture (Hochreiter and Schmidhuber, 1997) to model the dynamics of physical objects. Mittal et al. (2022) investigate routing mechanisms for Mixture-of-Experts models. They find that specialisation can yield better results as the number of tasks increases. However, the learned routing strategies do not capture domain specialisation. In particular, approaches based on backpropagation to the task loss often collapse to a single module. Mittal et al. (2022)'s experiments are restricted to small models and synthetic binary classification and regression tasks; we study a novel routing method via vector quantisation and perform our experiments on architectures over 1B parameters on reasoning tasks.

Causal Models Causal models aim to answer queries requiring knowledge of the causal relationships linking the data (Bareinboim et al., 2022). Schölkopf et al. (2021); Goyal and Bengio (2020) argue that for artificial systems to achieve robust and o.o.d reasoning, they must reason in terms of causes and effects and not only correlations, which current LLMs cannot do yet (Bareinboim et al., 2022; Zecevic et al., 2023). Structural Causal Models (SCMs) are graphical models representing causal relationships as mapping functions from parent nodes to their child nodes in a Directed Acyclic Graph (Pearl, 2009). If fully specified, an SCM can represent the complete inner workings of a system. However, building an SCM requires access to high-level causal variables, which is not the case in many deep learning tasks that take low-level observations as inputs (Schölkopf et al., 2021). The **do-calculus**,

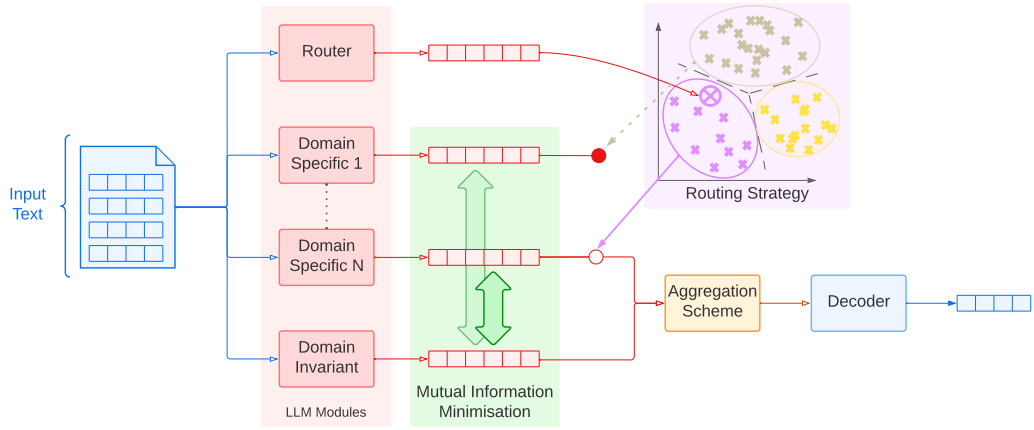


Figure 1: Proposed Independent Causal Language Models (ICLM) architecture for language-modelling tasks. The input text (on the left, in blue) is fed to multiple pretrained LLM modules (in red). A router uses clustering on input text embeddings (in purple) to activate a domain-specific module for this input. The domain-invariant module is always activated. The latent representations generated by the activated modules are combined using an aggregation scheme (in orange) and converted into a probability distribution for the next word (on the right, in blue). An additional loss (in green) minimises the Mutual Information between the domain-invariant and the domain-specific representations. The router ensures that the domain-specific modules only gain in-domain knowledge while the Mutual Information loss regularises the domain-invariant module towards learning abstract representations.

defined by Pearl (1995, 2009), is used to identify the causal effect of a variable on another with the help of the **do** operator: $\mathbf{do}(\cdot)$ represents an intervention, i.e. the forced attribution of a value to a variable. If $P(Y|\mathbf{do}(X)) = P(Y)$, then X has no causal effect on Y (they may still be correlated if they share common ancestors). Another class of causal models relies on determining the flow of *information* in a system (Shannon, 1948; Paluš et al., 2001; Schreiber, 2000). However, these concepts have yet to be applied to language models. In the domain of transformers, the Causal Transformer (Melnychuk et al., 2022) and Causal Attention (Yang et al., 2021) introduce cross-attention mechanisms to reduce biases from the training distribution.

3 Causal Information Routing for LLMs

We now describe our proposed modular architecture: *Independent Causal Language Models* (ICLM), where each module is an LLM fine-tuned for a specific specialisation or generalisation objective. We aim to build a system that can adapt to changing distributions and capture better abstractions. Our architecture is separated into $N + 2$ **LLM modules** connected by three main **components**. The LLM modules are composed of a *router* that generates embeddings of the inputs, a *domain-invariant module* trained to learn abstractions and N *domain-specific modules* trained to specialise on a single task or domain. The other components are the *routing strategy*, *Mutual Information loss* and the *aggregation scheme*. The *routing strategy* uses the embeddings from the LLM router to redirect the inputs to a specific module. Specifically, routing is performed in an unsupervised fashion: each embedding is projected into a clustered space. The centroid of each cluster is associated with a domain-specific module to which it assigns a binary activation weight for a given input. If the input belongs to a cluster, the corresponding module is activated. By contrast, the domain-invariant module processes all inputs. The *Mutual Information loss* induces abstraction within the domain-invariant module; minimising this loss reduces shared information between the domain-specific and domain-invariant modules. I.e. it is intended to cause the domain-invariant module to gain domain-invariant knowledge and the domain-specific modules domain-specific knowledge. Finally, the *aggregation scheme* combines the output of the activated domain-specific module and the domain-invariant module to produce the final output. Figure 1 shows an overview of our method. We describe the routing strategy in Section 3.1, the Mutual Information minimisation in Section 3.2 and the aggregation scheme in Section 3.3. Section 4 discusses how the architecture reflects Indepen-

dent Causal Mechanisms.

3.1 Routing Strategy

The routing strategy redirects the input tokens to a domain-specific module. This step divides the inference into independent modules to increase the specialisation of each module and reduce spurious distribution biases. In particular, the distribution may be imbalanced: a data class may dominate the training distribution and spuriously drive the gradients in a dense model. The routing module is used to balance out the distribution. The inputs belonging to the dominant class are restricted to a single module and cannot adversarially affect the other modules. In parallel, data points far from this data class are trained on a specialised module.

We use a pre-trained LLM (the router) with no final language modelling layer to build an input embedding space. The embeddings serve as inputs to the unsupervised routing strategy. In the strategy, all modules receive the inputs, but the outputs of non-activated modules are *blocked*. I.e. their outputs are associated with a weight of zero (activated modules have a weight equal to one). This activation process by weighting allows us to study more complex (non-binary) weighting schemes, discussed in Appendix C. This unsupervised learning method grants more flexibility than the matrix multiplication used in sparse transformers, as any clustering algorithm can be used. In particular, in continual learning settings, one could imagine using a varying number of clusters (Ester et al., 1996) and dynamically allocating new modules as data is being fed to the router. In our work, we restrict ourselves to simple clustering methods as we find that they are sufficiently fine-grained for our tasks. We perform clustering at the input level, i.e. each point in the clustering space represents a complete input context.

Vector Quantisation We use the vector quantisation procedure introduced for the VQ-VAE (van den Oord et al., 2017) as a clustering method. N vectors h_c are arbitrarily initialised in the embedding space, acting as cluster centroids. The attribution of an input to a cluster is determined by measuring the shortest Euclidean distance between them. The router generates an embedding for each token in the input so we measure the distance between a centroid and each token and sum them to obtain the total distance. The location of the centroids is iteratively updated to move closer to the input embeddings using vector quantisation. The corresponding routing loss is defined as follows:

$$\mathcal{L}_R = \text{MSE}(sg(h_c), h_r) + \nu \cdot \text{MSE}(h_c, sg(h_r)) \quad (1)$$

with h_r one token embedding and h_c the coordinates of the selected centroid, sg is the *stop_gradients* operation, and ν is a hyperparameter. This method has been very successful in transposing high-level concepts from a continuous to a discrete space (Bao et al., 2022; Ramesh et al., 2021) and in building disentangled or interpretable semantic spaces (Gendron et al., 2023b; Yu et al., 2023). This approach is simple and assumes clusters with non-overlapping convex hulls. We consider more complex strategies in Appendix C.

3.2 Mutual Information Minimisation

The second aim of the architecture is to induce abstraction and domain-invariance in LLMs. To this end, we introduce a regularisation process based on information theory. We minimise the Mutual Information (I) (Shannon, 1948; Kreer, 1957) between the domain-specific and domain-invariant modules. Specifically, we minimise the information between the last hidden states of the modules. The idea is to drive the domain-specific modules to gain knowledge specific to their distribution only, while the domain-invariant module gains knowledge common to all distributions and discards the domain-specific information that could be detrimental to generalisation. The Mutual Information between two random processes corresponds to the dependence between the two processes, i.e. the amount of information gained on the first process by observing the second one. The Mutual Information between two random variables $H_I \in \mathcal{H}$ and $H_S \in \mathcal{H}$ is given by:

$$\text{I}(H_I, H_S) = \text{KL}(P_{H_I,H_S} || P_{H_I} \otimes P_{H_S}) \quad (2)$$

where KL is the Kullback-Leibler divergence (Kullback and Leibler, 1951), H_I is the random variable representing the last hidden state of the domain-invariant module and H_S is its counterpart in one domain-specific module. The hidden states are interpreted as logits distributed in a feature space $\mathcal{H}$. P_{H_I,H_S} is their joint distribution, and P_{H_I} and P_{H_S} are their marginals. They are later decoded using a final linear layer into the space of possible next words corresponding to the vocabulary of the

LLM. The total loss $\mathcal{L}_{\text{I}}$ is given by the total information shared between the domain-invariant and all N domain-specific modules:

$$\mathcal{L}_{\text{I}} = \sum_{n \in [1,N]} \text{I}(H_I, H_{S_n}) \quad (3)$$

The probabilities $P_{H_I}(h)$ and $P_{H_S}(h)$ cannot be directly computed for any given hidden state $h \in \mathcal{H}$. We can only access the probabilities $P_{H_I}(h|c)$ and $P_{H_S}(h|c)$ for a given input context $c \in \mathcal{C}$. The marginalisation on $\mathcal{C}$ is intractable because of the exponential input space: $|\mathcal{C}| = V^L$, with V the vocabulary size of the LLM and L the maximum length of the input sequence (typically $V^L = (32.10^3)^{4096}$). We can approximate it by sampling $\mathcal{C}$ at the batch level $\mathcal{B}$: $P(h) = \sum_{c \in \mathcal{C}} P(h|c) \cdot P(c) \approx \frac{1}{|\mathcal{B}|} \sum_{c \in \mathcal{B}} P(h|c)$ with $|\mathcal{B}| \ll |\mathcal{C}|$. We do the same with the joint distribution P_{H_I,H_S}.

3.3 Aggregation of Outputs

Before aggregating the domain-invariant and domain-specific modules, we perform a shared batch normalisation (Ioffe and Szegedy, 2015) between their last hidden states. For a batch of size $|\mathcal{B}|$, one domain-specific active module and one domain-invariant module, batch normalisation is operated on $2 \times |\mathcal{B}|$ samples. Batch normalisation ensures that the module outputs have the same mean and variance. We then use a standard language modelling head that converts the hidden states into a probability distribution for the next token. The language modelling head is a fully connected layer that takes the concatenated hidden states as inputs and outputs a probability distribution in the vocabulary of the language model. This is a simple aggregation method with great expressivity due to the shared final dense layer. However, this layer can be subject to biases, e.g. if prioritising information from one module at the expense of the others. We study other aggregation schemes, less expressive but more resilient to this issue, in Appendix D.

Loss Function The total training loss is composed of five components:

$$\mathcal{L} = \mathcal{L}_o + \alpha \cdot \mathcal{L}_{inv} + \beta \cdot \mathcal{L}_{dom} + \gamma \cdot \mathcal{L}_R + \epsilon \cdot \mathcal{L}_{\text{I}} \quad (4)$$

$\mathcal{L}_o$ is the self-supervised cross-entropy loss between the output logits of ICLM and the target text.

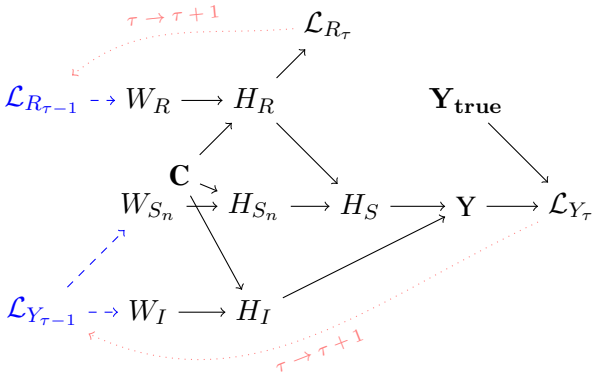


Figure 2: Simplified temporal causal graph $\mathcal{G}$ during training before adding Mutual Information minimisation. C is the input context. H_R, H_I, H_{S_n}, H_S are the latent states of the router, domain-invariant, domain-specific and activated domain-specific (after router weighting) modules. For simplicity, we only show the state H_{S_n} of the activated domain-specific module n. Y and Y_{true} are the output and true distributions. W_R, W_{S_n} and W_I are the trainable parameters of the modules. $\mathcal{L}_Y = \mathcal{L}_o + \alpha \cdot \mathcal{L}_{inv} + \beta \cdot \mathcal{L}_{dom}$ and $\mathcal{L}_R$ are the output and router losses. Black edges show the forward pass at step τ. Blue dashed edges show the backward pass at step τ. Red dotted edges illustrate the causal links between the forward and backward passes.

$\mathcal{L}_{inv}$ and $\mathcal{L}_{dom}$ are cross-entropy losses between the output logits of the invariant module and those of the activated domain-specific module. $\mathcal{L}_R$ is the vector quantisation loss obtained from the routing strategy (Eq. 1). $\mathcal{L}_{\text{I}}$ is the Mutual Information loss (Eq. 3). We consider three separate self-supervised losses $\mathcal{L}_o$, $\mathcal{L}_{inv}$ and $\mathcal{L}_{dom}$ to induce the modules to match the target distributions individually and prevent collapse to a single useful module. α, β, γ and ϵ are constant hyperparameters.

4 Theoretical Perspective

In this section, we provide theoretical evidence on how our model approximates Independent Causal Mechanisms and under what assumptions. Independent Causal Mechanisms consist of autonomous modules that work independently. In our case, all domain-specific modules are trained for specific tasks/distributions. The domain-invariant module is trained only to use domain-invariant knowledge. The router module is tasked to split the input distribution into N more balanced distributions. We aim to verify that the modules are not causally related. More formally, we aim to study under what conditions the following holds:

$$P(H_R|\mathbf{do}(H_{S_n})) = P(H_R)\ \forall n \in [1, N] \quad (5)$$

$$P(H_R|\mathbf{do}(H_I)) = P(H_R) \quad (6)$$

$$P(H_{S_n}|\mathbf{do}(H_{S_{\hat{n}}})) = P(H_{S_n}) \quad \forall \hat{n} \in [1, N] \setminus \{n\}\ \forall n \in [1, N] \quad (7)$$

$$P(H_I|\mathbf{do}(H_{S_n})) = P(H_I)\ \forall n \in [1, N] \quad (8)$$

H_R, H_I and H_{S_n} $\forall n \in [1, N]$ are the respective representations generated by the router, domain-invariant and N domain-specific modules.

Equations 5 and 6 are verified. The proof is provided in Appendix B; the main idea is that the use of a separate loss function for training the router prevents the other modules from causally acting on the router, either in the forward or backward passes. It can be verified in Figure 2. However, if an invariant module is part of the model, Equations 7 and 8 do not hold. The domain-specific modules do not directly influence each other because the routing mechanism allows a single module to go through the forward and backward passes. Nevertheless, a causal path can be drawn through the domain-invariant module as it is always activated. For example, assuming a model with two domain-specific modules, S_0 and S_1, activated one after the other, a path exists and can be represented in a simplified version as $H_{S_1} \dashrightarrow \mathcal{L}_{Y_\tau} \dashrightarrow H_I \dashrightarrow \mathcal{L}_{Y_{\tau+1}} \dashrightarrow H_{S_2}$. Again, details of the proof are given in Appendix B. As H_I and H_{S_n} are causally related, we need to reduce the dependency between the two quantities using a regularisation term. Minimising the Mutual Information between H_I and H_{S_n} amounts to reducing the mutual dependence between the variables. $I(H_I, H_{S_n}) = 0$ if and only if H_I and H_{S_n} are independent. If verified, the loss $\mathcal{L}_Y$ can be divided into two independent components and Equations 7 and 8 hold. We verify experimentally in Appendix E.1 that the Mutual Information is close to zero after $\sim$ 50 training steps.

5 Abstract Reasoning with ICLM

5.1 Experimental Setup

By default, we use $N = 2$ domain-specific modules and one domain-invariant module, as the datasets we use contain two subdomains each. We also perform experiments with an ablated model that does not have a domain-invariant module. In addition, we study the individual performance of the domain-invariant and domain-specific modules. We use a pretrained LLaMA2-7B (Touvron et al., 2023b) for all our modules. We use Low-Rank Approximation of LLMs (LoRA) (Hu et al., 2022) to fine-tune the modules on their respective tasks. All models are fine-tuned for 3 epochs with AdamW (Loshchilov and Hutter, 2019) and a batch size of 16. Loss hyperparameters are $\alpha = 0.1$, $\beta = 0.1$, $\gamma = 0.1$, $\epsilon = 0.01$, $\nu = 0.25$. It is worth noting that the number of parameters used is only marginally higher than that of the base LLaMA2, as only low-memory LoRA adapter weights are learned during training.

5.2 Datasets

We perform experiments on the text-based ACRE and RAVEN datasets (Zhang et al., 2021a, 2019; Gendron et al., 2023a)[1]. ACRE and RAVEN are adapted from Visual Question Answering datasets to be used by language models. The visual ACRE (Zhang et al., 2021a) is an abstract causal reasoning dataset where the model must deduce the causal mechanisms from a small set of image examples. The visual RAVEN (Zhang et al., 2019) is an abstract reasoning dataset where the model must complete a sequence of Raven Progressive Matrices (Raven, 1938). The text ACRE and RAVEN contain symbolic and natural language descriptions of the images and instructions for solving the task. They require knowledge of the underlying causal mechanisms to be solved and have o.o.d sets to challenge this ability in the tested systems. More details about the datasets are provided in Appendix A.

Out-of-Distribution Regimes Each dataset has two o.o.d regimes. In ACRE, the *compositionality* split changes the composition of the context examples: combinations of figure shapes and colours unseen in the training set are proposed; the *systematicity* split alters the distribution of the context example activations: the context contains more positive examples than in the training set. In RAVEN, the *four* split contains four figures instead of one; the *in-center* split describes two figures with one containing the other instead of being placed next to each other.

5.3 Abstract and Causal Reasoning

The results obtained on the ACRE and RAVEN test sets are shown in Table 1. The proposed ICLM can outperform the baseline, particularly on the

[1]We use the data provided at https://github.com/Strong-AI-Lab/Logical-and-abstract-reasoning.

most challenging o.o.d sets. Moreover, the performance of the individual domain-invariant modules highlights that the modules have learned more generalisable knowledge than with standard training. The domain-specific modules compete with the baselines trained on the corresponding specific domain, showing that the router accurately distributes the inputs to the right modules. The modules even outperform the oracle router on RAVEN in almost all settings. We investigate a potential reason for this phenomenon in Section 6.2.

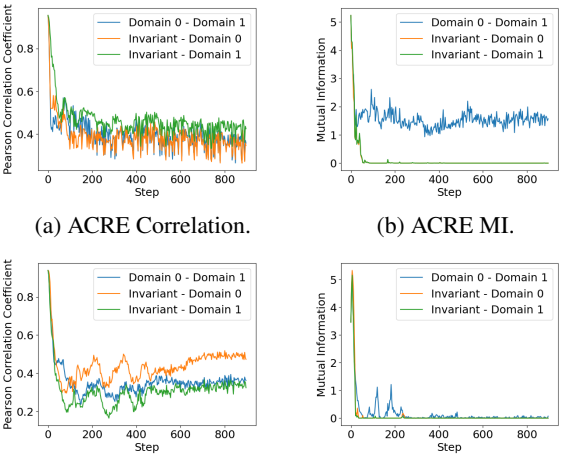
5.4 Continual Learning

We investigate the capacity of our model to be used in continual learning settings. Continual learning consists of training a model with continuous data streams or sets evolving over time, where the model acquires and accumulates knowledge incrementally. The main challenge lies in the *catastrophic forgetting* of the previous knowledge when gaining new information (Wang et al., 2023). We study a simple usecase where we want our model to learn one new task after training on a previous task. We choose the scenario $ACRE \rightarrow RAVEN$ as RAVEN is more challenging, particularly the o.o.d sets. The results are shown in Table 2. The domain-invariant module can use general information extracted from ACRE to improve its performance on RAVEN, even outperforming the baseline trained on RAVEN only. The domain-specific modules can also partially mitigate the catastrophic forgetting problem observed in LLaMA2. Their weights are not activated by the router on RAVEN inputs, thus not updated, and their performance on ACRE is preserved. However, the aggregation process is affected, leading to reduced performance on ACRE Text.

6 Routing and Independence Analysis

6.1 Evolution of Module Independence

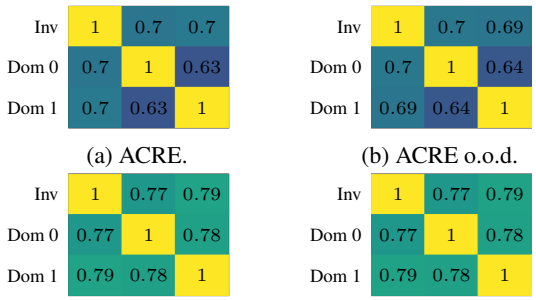
We study the independence of the module hidden states during training (Figure 3) and inference (Figure 4). we focus on two complementary measures: Mutual Information and the Pearson Correlation Coefficient that measures linear correlation between variables. The latter is limited to linear dependence but is more easily interpretable. The shared Mutual Information as well as the Pearson Correlation Coefficient between modules are effectively reduced by the regularisation scheme during fine-tuning. However, the module hidden states remain correlated, in particular at inference time. Further investigation in Appendix E.5 further shows that this correlation is maintained across most layers, which indicates the presence of a general domain-invariant mechanism shared by all modules and composing the basis of their reasoning abilities. Its influence is reduced via the fine-tuning procedure that develops domain-specific knowledge but it remains the main mechanism used, particularly at test time.



(a) ACRE Correlation. (b) ACRE MI.

(c) RAVEN Correlation. (d) RAVEN MI.

Figure 3: Evolution of independence measures between modules during fine-tuning on ACRE and RAVEN. We measure independence on the last hidden states of the modules. Correlation and MI are highly reduced but modules remain correlated.



(a) ACRE. (b) ACRE o.o.d.

(c) RAVEN. (d) RAVEN o.o.d.

Figure 4: Correlation between the last hidden states of the modules during inference at test time. Module states are more correlated than during training.

6.2 Routing Alignment

We look deeper at the embedding space in the routing module, projected into a 2D space using Multidimensional Scaling (MDS) (Borg and Groenen, 2005). Figure 5 shows the embedding spaces and their attribution to the domain-specific modules. Detailed attributions are shown in Appendix E.2.

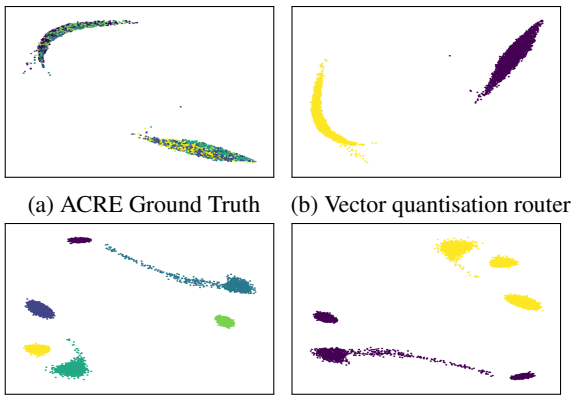
	ACRE		-o.o.d-Comp		-o.o.d-Sys		RAVEN		-o.o.d-Four		-o.o.d-In-Center	
	Text	Symb	Text	Symb	Text	Symb	Text	Symb	Text	Symb	Text	Symb
LLaMA2-Base	0.014	0.003	0.244	0.001	0.288	0.001	0.026	0.149	0.073	0.121	0.000	0.001
-Finetuned-All*	0.832	0.891	0.832	0.881	*0.911*	*0.891*	*0.990*	**1.000**	*0.673*	**0.743**	**0.673**	0.198
ICLM* (ours)	0.653	**0.950**	0.663	*0.931*	0.634	**0.901**	**1.000**	0.980	**0.703**	*0.703*	0.515	*0.228*
ICLM-No-Inv* (ours)	*0.871*	*0.921*	*0.842*	**0.941**	0.822	*0.891*	**1.000**	0.732	0.525	0.515	0.455	0.168
ICLM-Invariant* (ours)	**0.891**	*0.921*	**0.851**	**0.941**	**0.921**	*0.891*	**1.000**	*0.990*	0.634	0.693	*0.554*	**0.238**
ICLM-Domain* (ours)	0.871	0.911	0.822	0.901	0.822	0.891	0.980	0.980	0.604	0.634	0.386	0.228
-Finetuned-Oracle-Router	0.997	1.000	1.000	1.000	0.994	0.999	0.977	0.965	0.557	0.442	0.536	0.064

Table 1: Accuracy on the ACRE and RAVEN i.i.d and o.o.d test sets. "Finetuned-All" is a single LLaMA2 model fine-tuned on text and symbolic i.i.d training sets. "Finetuned-Oracle-Router" is an ensemble of two LLaMA2 models fine-tuned on each i.i.d training set (either text or symbolic) and routed via a ground-truth oracle. ICLM is our full model trained on text and symbolic i.i.d training sets. ICLM-Invariant shows the results for the domain-invariant module alone. ICML-Domain shows the results for the domain-specific module that aligns best with the dataset (see Appendix E.2). ICLM-No-Inv is an ablated ICLM with no domain-invariant module. Models with a * indicate that that this paper introduces the results. The best model is in **bold**, and the second best is in *italics*. ICLM outperform LLaMA2 on most sets and individual modules even outperform the oracle on the more challenging RAVEN.

	ACRE		-Comp		-Sys		RAVEN		-Four		-In-Center	
	Text	Symb	Text	Symb	Text	Symb	Text	Symb	Text	Symb	Text	Symb
ICLM$_{ACRE}$* (Table 1)	0.653	0.950	0.663	0.931	0.634	0.901	-	-	-	-	-	-
ICLM$_{RAVEN}$* (Table 1)	-	-	-	-	-	-	1.000	0.980	0.703	0.703	0.515	0.228
$ACRE \rightarrow RAVEN$												
ICLM* (ours)	*0.089*	**0.901**	0.119	**0.931**	0.050	**0.871**	**1.000**	**0.990**	**0.772**	**0.772**	**0.833**	**0.248**
ICLM-Invariant* (ours)	**0.287**	*0.396*	**0.277**	*0.416*	**0.238**	*0.455*	**1.000**	*0.970*	*0.673*	*0.723*	*0.723*	*0.238*
LLaMA2-Finetuned-Sequential*	0.079	0.376	*0.149*	0.386	*0.089*	0.426	*0.980*	0.772	0.634	0.554	0.584	0.069

Table 2: Accuracy when the model is trained sequentially on ACRE then RAVEN i.i.d training sets. ICLM can use the information from ACRE to improve its performance on RAVEN, outperforming the baseline trained on RAVEN only, while preserving more knowledge from the previous task than the base LLaMA2-7B finetuned sequentially. In particular, the accuracy of ACRE-Symbolic sets is almost untouched.

Both datasets have a clear division between text and symbolic embeddings. However, the o.o.d sets are not well separated in ACRE while they are in RAVEN. This division can explain the similarity in the results between the ACRE i.i.d and o.o.d sets, as shown in Table 1. Moreover, as the distributions are very similar, the impact of the router and the need for abstraction are reduced. On the other hand, there is a clear separation between the i.i.d and o.o.d RAVEN embeddings, explaining the differences in behaviours from the models across the sets. Adding more modules could allow taking more advantage of this separation, with each module specialising to a subdomain closer to one of the o.o.d embeddings.



(a) ACRE Ground Truth (b) Vector quantisation router

(c) RAVEN Ground Truth (d) Vector quantisation router

Figure 5: 2D projection of the hidden states of LLaMA2 on ACRE and RAVEN i.i.d and o.o.d sets. Ground truth samples are labelled as in Table 1 (text/symbolic i.i.d/o.o.d sets). Text and symbolic inputs are always clustered separately. i.i.d and o.o.d sets are clustered together in ACRE and separated in RAVEN. The router follows the text and symbolic division.

7 Conclusion

Performing strong out-of-distribution reasoning is a challenging task, and despite their impressive performance on a wide range of problems, LLMs have not demonstrated this ability yet. Combining this popular model with causal models could help bridge this gap. This work presents a modular architecture yielding LLMs to behave as Independent Causal Mechanisms. We show theoretically that the proposed model generates causally-independent modules. We perform experiments on abstract and causal reasoning tasks in o.o.d and continual learning settings and show that these principles increase strong reasoning and generalisation. We further show that the proposed modules specialise to their domain with fine-tuning but still partially rely on a shared domain-invariant mechanism, highlighting a limitation for representing ICMs with LLMs.

Limitations

The model proposed in this paper is constrained to work in a modular manner. All modules are sparsely connected at the level of the language modelling head. This single connection offers a useful initial framework to study the ICMs within the context of LLMs and can represent a wide range of problems (requiring the composition of several independent reasoning processes) but it can only represent causal DAGs with a single layer depth, potentially hindering the expressivity of more complex mechanism interactions. Generating complex causal computation graphs tailored to the task at hand may improve performance, but this problem is out of the scope of this paper.

We also focus our investigation on the independence and accuracy of the modules and do not attempt to directly represent the true causal mechanisms of the tasks as they are unknown. Moreover, we conduct experiments on reasoning tasks to verify if inducing high-level modularity can yield increases in performance and generalisation. We aim not to outperform the state-of-the-art on the problems but to study whether the proposed mechanisms can yield such increases.

In addition, training and fine-tuning Large Language Models has a high computational cost. Due to this high cost, we perform a single fine-tuning run per task and conduct experiments on this model. The training cost for our model is only slightly higher than for a base LLaMA2 because a quantized version of the base model can be used. However, the cost increases during inference because each module has to be associated with a fully loaded LLaMA2. Our current implementation loads all modules in parallel with the aim to study the interactions between them (see Section 6 and Appendix D), but this choice is memory intensive and does not permit us to directly build ICLM with a significantly larger number of modules. To scale ICLM, a production implementation could use a single base model and load only the required modules sequentially. Such implementation would have the same theoretical properties at a negligible memory cost.

References

Peter C. Austin. 2011. An introduction to propensity score methods for reducing the effects of confounding in observational studies. *Multivariate Behavioral Research*, 46(3):399–424. PMID: 21818162.

Hangbo Bao, Li Dong, Songhao Piao, and Furu Wei. 2022. Beit: BERT pre-training of image transformers. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net.

Qiming Bao, Gaël Gendron, Alex Yuxuan Peng, Wanjun Zhong, Neset Tan, Yang Chen, Michael Witbrock, and Jiamou Liu. 2023. A systematic evaluation of large language models on out-of-distribution logical reasoning tasks. *CoRR*, abs/2310.09430.

Elias Bareinboim, Juan D. Correa, Duligur Ibeling, and Thomas Icard. 2022. On pearl's hierarchy and the foundations of causal inference. In Hector Geffner, Rina Dechter, and Joseph Y. Halpern, editors, *Probabilistic and Causal Inference: The Works of Judea Pearl*, volume 36 of *ACM Books*, pages 507–556. ACM.

Ingwer Borg and Patrick JF Groenen. 2005. *Modern multidimensional scaling: Theory and applications*. Springer Science & Business Media.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*, NIPS'20, Red Hook, NY, USA. Curran Associates Inc.

Sébastien Bubeck, Varun Chandrasekaran, Ronen Eldan, Johannes Gehrke, Eric Horvitz, Ece Kamar, Peter Lee, Yin Tat Lee, Yuanzhi Li, Scott Lundberg, Harsha Nori, Hamid Palangi, Marco Tulio Ribeiro, and Yi Zhang. 2023. Sparks of artificial general intelligence: Early experiments with gpt-4.

François Chollet. 2019. On the measure of intelligence. *CoRR*, abs/1911.01547.

Aidan Clark, Diego de Las Casas, Aurelia Guy, Arthur Mensch, Michela Paganini, Jordan Hoffmann, Bogdan Damoc, Blake A. Hechtman, Trevor Cai, Sebastian Borgeaud, George van den Driessche, Eliza Rutherford, Tom Hennigan, Matthew J. Johnson, Albin Cassirer, Chris Jones, Elena Buchatskaya, David Budden, Laurent Sifre, Simon Osindero, Oriol Vinyals, Marc'Aurelio Ranzato, Jack W. Rae, Erich Elsen, Koray Kavukcuoglu, and Karen Simonyan. 2022. Unified scaling laws for routed language models. In *International Conference on Machine Learning, ICML 2022, 17-23 July 2022, Baltimore, Maryland, USA*, volume 162 of *Proceedings of Machine Learning Research*, pages 4057–4086. PMLR.

Martin Ester, Hans-Peter Kriegel, Jörg Sander, and Xiaowei Xu. 1996. A density-based algorithm for discovering clusters in large spatial databases with noise. In *kdd*, volume 96, pages 226–231.

William Fedus, Jeff Dean, and Barret Zoph. 2022a. A review of sparse expert models in deep learning. *CoRR*, abs/2209.01667.

William Fedus, Barret Zoph, and Noam Shazeer. 2022b. Switch transformers: Scaling to trillion parameter models with simple and efficient sparsity. *J. Mach. Learn. Res.*, 23:120:1–120:39.

Gaël Gendron, Qiming Bao, Michael Witbrock, and Gillian Dobbie. 2023a. Large language models are not abstract reasoners. *CoRR*, abs/2305.19555.

Gaël Gendron, Michael Witbrock, and Gillian Dobbie. 2023b. Disentanglement of latent representations via causal interventions. In *Proceedings of the Thirty-Second International Joint Conference on Artificial Intelligence, IJCAI 2023, 19th-25th August 2023, Macao, SAR, China*, pages 3239–3247. ijcai.org.

Gaël Gendron, Michael Witbrock, and Gillian Dobbie. 2023c. A survey of methods, challenges and perspectives in causality. *CoRR*, abs/2302.00293.

Anirudh Goyal and Yoshua Bengio. 2020. Inductive biases for deep learning of higher-level cognition. *CoRR*, abs/2011.15091.

Anirudh Goyal, Alex Lamb, Jordan Hoffmann, Shagun Sodhani, Sergey Levine, Yoshua Bengio, and Bernhard Schölkopf. 2021. Recurrent independent mechanisms. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net.

Suchin Gururangan, Mike Lewis, Ari Holtzman, Noah A. Smith, and Luke Zettlemoyer. 2022. Demix layers: Disentangling domains for modular language modeling. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL 2022, Seattle, WA, United States, July 10-15, 2022*, pages 5557–5576. Association for Computational Linguistics.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural Comput.*, 9(8):1735–1780.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2022. Lora: Low-rank adaptation of large language models. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net.

Sergey Ioffe and Christian Szegedy. 2015. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *Proceedings of the 32nd International Conference on Machine Learning, ICML 2015, Lille, France, 6-11 July 2015*, volume 37 of *JMLR Workshop and Conference Proceedings*, pages 448–456. JMLR.org.

Albert Q. Jiang, Alexandre Sablayrolles, Antoine Roux, Arthur Mensch, Blanche Savary, Chris Bamford, Devendra Singh Chaplot, Diego de Las Casas, Emma Bou Hanna, Florian Bressand, Gianna Lengyel, Guillaume Bour, Guillaume Lample, Lélio Renard Lavaud, Lucile Saulnier, Marie-Anne Lachaux, Pierre Stock, Sandeep Subramanian, Sophia Yang, Szymon Antoniak, Teven Le Scao, Théophile Gervet, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. 2024. Mixtral of experts. *CoRR*, abs/2401.04088.

Zhijing Jin, Jiarui Liu, Zhiheng Lyu, Spencer Poff, Mrinmaya Sachan, Rada Mihalcea, Mona T. Diab, and Bernhard Schölkopf. 2023. Can large language models infer causation from correlation? *CoRR*, abs/2306.05836.

J. Kreer. 1957. A question of terminology. *IRE Transactions on Information Theory*, 3(3):208–208.

Solomon Kullback and Richard A Leibler. 1951. On information and sufficiency. *The annals of mathematical statistics*, 22(1):79–86.

Hanmeng Liu, Ruoxi Ning, Zhiyang Teng, Jian Liu, Qiji Zhou, and Yue Zhang. 2023. Evaluating the logical reasoning ability of chatgpt and GPT-4. *CoRR*, abs/2304.03439.

Stuart Lloyd. 1982. Least squares quantization in pcm. *IEEE transactions on information theory*, 28(2):129–137.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net.

Valentyn Melnychuk, Dennis Frauen, and Stefan Feuerriegel. 2022. Causal transformer for estimating counterfactual outcomes. In *International Conference on Machine Learning, ICML 2022, 17-23 July 2022, Baltimore, Maryland, USA*, volume 162 of *Proceedings of Machine Learning Research*, pages 15293–15329. PMLR.

Sarthak Mittal, Yoshua Bengio, and Guillaume Lajoie. 2022. Is a modular architecture enough? In *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS 2022, New Orleans, LA, USA, November 28 - December 9, 2022*.

Milan Paluš, Vladimír Komárek, Zbyněk Hrnčíř, and Katalin Štěrbová. 2001. Synchronization as adjustment of information rates: Detection from bivariate time series. *Physical Review E*, 63(4):046211.

Judea Pearl. 1988. *Probabilistic reasoning in intelligent systems: networks of plausible inference*. Morgan kaufmann.

Judea Pearl. 1995. Causal diagrams for empirical research. *Biometrika*, 82(4):669–688.

Judea Pearl. 2009. *Causality*. Cambridge university press.

Jonas Peters, Dominik Janzing, and Bernhard Schölkopf. 2017. *Elements of causal inference: foundations and learning algorithms*. The MIT Press.

Aditya Ramesh, Mikhail Pavlov, Gabriel Goh, Scott Gray, Chelsea Voss, Alec Radford, Mark Chen, and Ilya Sutskever. 2021. Zero-shot text-to-image generation. In *Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event*, volume 139 of *Proceedings of Machine Learning Research*, pages 8821–8831. PMLR.

John C Raven. 1938. Raven standard progressive matrices. *Journal of Cognition and Development*.

Bernhard Schölkopf, Francesco Locatello, Stefan Bauer, Nan Rosemary Ke, Nal Kalchbrenner, Anirudh Goyal, and Yoshua Bengio. 2021. Towards causal representation learning. *CoRR*, abs/2102.11107.

Thomas Schreiber. 2000. Measuring information transfer. *Physical review letters*, 85(2):461.

Claude Elwood Shannon. 1948. A mathematical theory of communication. *The Bell system technical journal*, 27(3):379–423.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurélien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. 2023a. Llama: Open and efficient foundation language models. *CoRR*, abs/2302.13971.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Dan Bikel, Lukas Blecher, Cristian Canton-Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, Brian Fuller, Cynthia Gao, Vedanuj Goswami, Naman Goyal, Anthony Hartshorn, Saghar Hosseini, Rui Hou, Hakan Inan, Marcin Kardas, Viktor Kerkez, Madian Khabsa, Isabel Kloumann, Artem Korenev, Punit Singh Koura, Marie-Anne Lachaux, Thibaut Lavril, Jenya Lee, Diana Liskovich, Yinghai Lu, Yuning Mao, Xavier Martinet, Todor Mihaylov, Pushkar Mishra, Igor Molybog, Yixin Nie, Andrew Poulton, Jeremy Reizenstein, Rashi Rungta, Kalyan Saladi, Alan Schelten, Ruan Silva, Eric Michael Smith, Ranjan Subramanian, Xiaoqing Ellen Tan, Binh Tang, Ross Taylor, Adina Williams, Jian Xiang Kuan, Puxin Xu, Zheng Yan, Iliyan Zarov, Yuchen Zhang, Angela Fan, Melanie Kambadur, Sharan Narang, Aurélien Rodriguez, Robert Stojnic, Sergey Edunov, and Thomas Scialom. 2023b. Llama 2: Open foundation and fine-tuned chat models. *CoRR*, abs/2307.09288.

Aäron van den Oord, Oriol Vinyals, and Koray Kavukcuoglu. 2017. Neural discrete representation learning. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 6306–6315.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 5998–6008.

Liyuan Wang, Xingxing Zhang, Hang Su, and Jun Zhu. 2023. A comprehensive survey of continual learning: Theory, method and application. *CoRR*, abs/2302.00487.

Jason Wei, Maarten Paul Bosma, Vincent Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew Mingbo Dai, and Quoc V. Le. 2022a. Finetuned language models are zero-shot learners.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, brian ichter, Fei Xia, Ed Chi, Quoc V Le, and Denny Zhou. 2022b. Chain-of-thought prompting elicits reasoning in large language models. In *Advances in Neural Information Processing Systems*, volume 35, pages 24824–24837. Curran Associates, Inc.

Zhaofeng Wu, Linlu Qiu, Alexis Ross, Ekin Akyürek, Boyuan Chen, Bailin Wang, Najoung Kim, Jacob Andreas, and Yoon Kim. 2023. Reasoning or reciting? exploring the capabilities and limitations of language models through counterfactual tasks. *CoRR*, abs/2307.02477.

Xu Yang, Hanwang Zhang, Guojun Qi, and Jianfei Cai. 2021. Causal attention for vision-language tasks. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2021, virtual, June 19-25, 2021*, pages 9847–9857. Computer Vision Foundation / IEEE.

Lijun Yu, Yong Cheng, Zhiruo Wang, Vivek Kumar, Wolfgang Macherey, Yanping Huang, David A. Ross, Irfan Essa, Yonatan Bisk, Ming-Hsuan Yang, Kevin Murphy, Alexander G. Hauptmann, and Lu Jiang. 2023. SPAE: semantic pyramid autoencoder for multimodal generation with frozen llms. *CoRR*, abs/2306.17842.

Matej Zecevic, Moritz Willig, Devendra Singh Dhami, and Kristian Kersting. 2023. Causal parrots: Large language models may talk causality but are not causal. *CoRR*, abs/2308.13067.

Chi Zhang, Feng Gao, Baoxiong Jia, Yixin Zhu, and Song-Chun Zhu. 2019. RAVEN: A dataset for relational and analogical visual reasoning. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA, June 16-20, 2019*, pages 5317–5327. Computer Vision Foundation / IEEE.

Chi Zhang, Baoxiong Jia, Mark Edmonds, Song-Chun Zhu, and Yixin Zhu. 2021a. ACRE: abstract causal reasoning beyond covariation. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2021, virtual, June 19-25, 2021*, pages 10643–10653. Computer Vision Foundation / IEEE.

Chiyuan Zhang, Maithra Raghu, Jon M. Kleinberg, and Samy Bengio. 2021b. Pointer value retrieval: A new benchmark for understanding the limits of neural network generalization. *CoRR*, abs/2107.12580.

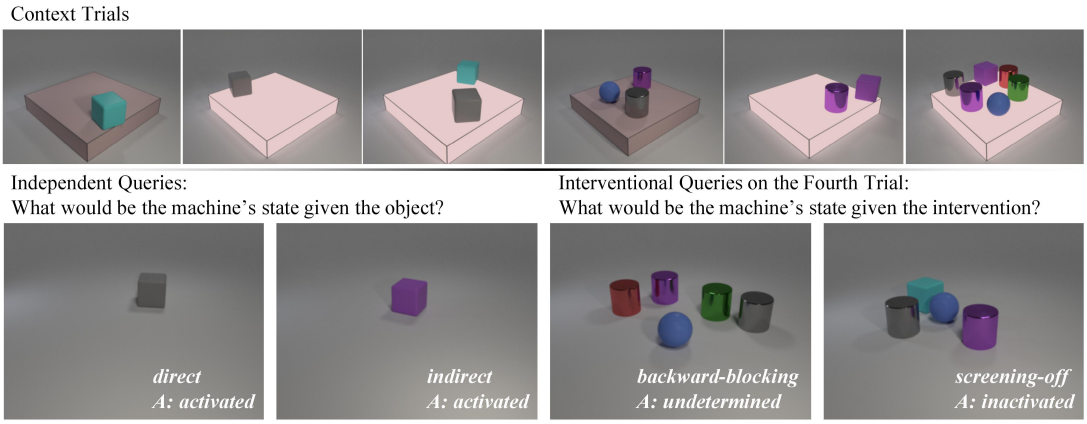


Figure 6: Example of ACRE task, from the original visual dataset (Zhang et al., 2021a).

A Dataset Description

We perform experiments on the ACRE (Zhang et al., 2021a) and RAVEN (Zhang et al., 2019) datasets adapted to text format to be used by language models by (Gendron et al., 2023a). We use the data provided at https://github.com/Strong-AI-Lab/Logical-and-abstract-reasoning.

A.1 ACRE

ACRE is an abstract causal reasoning dataset where the model must deduce the causal mechanisms from a small set of image examples. Each sample in the dataset contains six images representing objects and a light, activated or not. The goal of the task is to determine from the images the objects causing the activation of the light and determine the state of the light in four test cases: activated, deactivated, or undetermined if the activation causes cannot be retrieved. Figures 6, 7 and 8 provide examples from the datasets.

Pre-Prompt

Objects of various color, shape, and texture are displayed. Some objects may contain a device to turn a light on if displayed. From the observations, deduce if the light is on, off, or if the state cannot be determined. Your answer must contain a single word:
on.
off.
undetermined.

Example Cases

A cyan cylinder in rubber is visible. The light is on.
A gray cube in rubber is visible. The light is off.
A cyan cylinder in rubber is visible. A gray cube in rubber is visible. The light is on.
A blue cube in metal is visible. The light is off.
A gray cylinder in rubber is visible. A gray cube in metal is visible. The light is off.
A red sphere in metal is visible. A yellow cube in rubber is visible. The light is on.

Test Case

A red sphere in metal is visible. The light is *undetermined*

Figure 7: Example of natural language ACRE task, from (Gendron et al., 2023a). In the test case, the target answer is indicated in *italics*.

Pre-Prompt

Figure out the pattern in the following examples and apply it to the test case. Your answer must follow the format of the examples. You can answer 1 if the solution cannot be determined. Your answer must be one of the following choices:
0.
1.
2.

Example Cases

[28] → 2
[0] → 0
[28, 0] → 2
[5] → 0
[16, 1]→ 0
[35, 14] → 2

Test Case

[35] → *1*

Figure 8: Example of symbolic ACRE task, from (Gendron et al., 2023a). In the test case, the target answer is indicated in *italics*.

A.2 RAVEN

RAVEN is an abstract reasoning dataset where the model must complete a sequence of Raven Progressive Matrices (Raven, 1938). Each sample in the dataset contains eight Raven Progressive Matrices (Raven, 1938). The goal of the task is to determine the matrix that completes the sequence from a set of eight propositions. Figures 9 and 10 provide examples from the datasets. The RAVEN dataset contains multiple splits, with different categories of matrices: with a single figure, with four figures, with nine figures, with two figures side by side, with two figures up and down, with one figure inside another, an with four figures inside another one. In our experiments, we use the set with a single figure for training and i.i.d testing and the sets with four and nine figures for o.o.d testing. Figure 9 shows two examples from the set with four figures and from the set with four figures inside one.

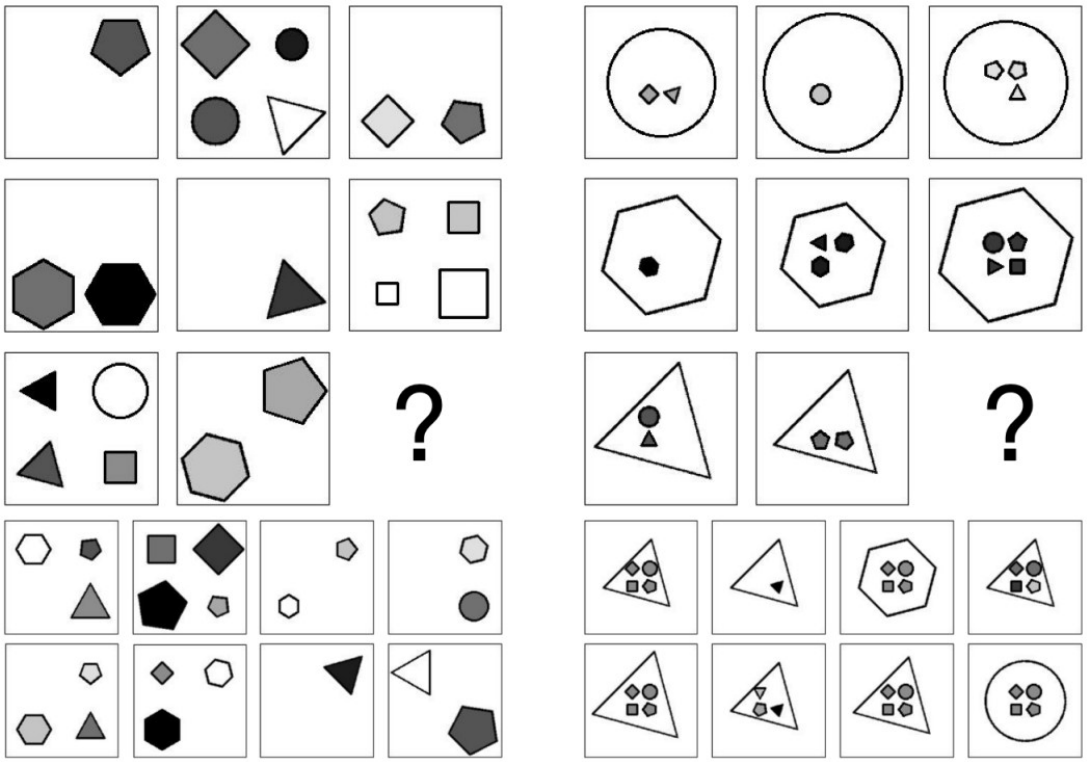


Figure 9: Example of RAVEN tasks, from the original visual dataset (Zhang et al., 2021a).

Pre-Prompt

Find the pattern number 9 that completes the sequence. Pick the letter in front of the correct pattern that logically follows in the sequence from the answer set. Patterns in the sequence are preceded by a number from 1 to 8. Patterns in the answer set are preceded by a letter from A to H. Only return the letter in front of the correct pattern.

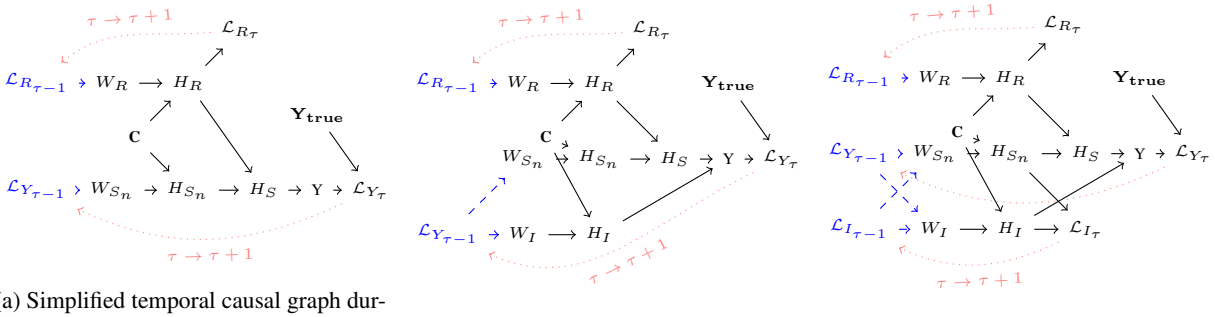
Example Cases

1. [(D, D, C, H,)]	1. On an image, a large lime square rotated at 180 degrees.
2. [(C, D, C, H,)]	2. On an image, a medium lime square rotated at 180 degrees.
3. [(E, D, C, H,)]	3. On an image, a huge lime square rotated at 180 degrees.
4. [(E, C, F, D,)]	4. On an image, a huge yellow circle rotated at 0 degrees.
5. [(D, C, F, D,)]	5. On an image, a large yellow circle rotated at 0 degrees.
6. [(C, C, F, D,)]	6. On an image, a medium yellow circle rotated at 0 degrees.
7. [(C, J, E, B,)]	7. On an image, a medium white hexagon rotated at -90 degrees.
8. [(E, J, E, B,)]	8. On an image, a huge white hexagon rotated at -90 degrees.
A. [(A, J, E, B,)]	A. On an image, a tiny white hexagon rotated at -90 degrees.
B. [(F, J, E, B,)]	B. On an image, a giant white hexagon rotated at -90 degrees.
C. [(D, A, E, B,)]	C. On an image, a large red hexagon rotated at -90 degrees.
D. [(D, B, E, B,)]	D. On an image, a large orange hexagon rotated at -90 degrees.
E. [(D, J, E, B,)]	E. On an image, a large white hexagon rotated at -90 degrees.
F. [(D, E, E, B,)]	F. On an image, a large green hexagon rotated at -90 degrees.
G. [(D, G, E, B,)]	G. On an image, a large blue hexagon rotated at -90 degrees.
H. [(D, C, E, B,)]	H. On an image, a large yellow hexagon rotated at -90 degrees.

Test Case

The answer is *E*

Figure 10: Example of RAVEN task, from (Gendron et al., 2023a). In the test case, the target answer is indicated in *italics*. The text in blue shows the text for the symbolic dataset. The text in green shows the text for the natural language dataset. The text in gray is the same for both datasets.



(a) Simplified temporal causal graph during training with no domain-invariant module.

(b) Simplified temporal causal graph $\mathcal{G}$. The graph is the same as Figure 2.

(c) Temporal causal graph with Information Minimisation loss added.

Figure 11: Causal graphs with and without domain-invariant module and Mutual Information minimisation loss. C is the input context. H_R, H_I, H_{S_n}, H_S are the latent states of the router, domain-invariant, domain-specific and activated domain-specific (after router weighting) modules. For simplicity, we only show the state H_{S_n} of the activated domain-specific module n. Y and Y_{true} are the output and true distributions. W_R, W_{S_n} and W_I are the trainable parameters of the modules. $\mathcal{L}_Y = \mathcal{L}_o + \alpha \cdot \mathcal{L}_{inv} + \beta \cdot \mathcal{L}_{dom}$ and $\mathcal{L}_R$ are the output and router losses. Black edges show the forward pass at step τ. Blue dashed edges show the backward pass at step τ. Red dotted edges illustrate the causal links between the forward and backward passes. For simplicity, we only show the step for the loss variables as they appear twice. All other variables are at step τ.

B Supplement to the Theoretical Perspective

In this section, we prove the assertions made in Section 4 of the main paper. We study under what conditions the following equations (repeated from Section 4) hold:

$$P(H_R|\mathbf{do}(H_{S_n})) = P(H_R)\ \forall n \in [1, N] \tag{9}$$

$$P(H_R|\mathbf{do}(H_I)) = P(H_R) \tag{10}$$

$$P(H_{S_n}|\mathbf{do}(H_{S_{\hat{n}}})) = P(H_{S_n}) \quad \forall \hat{n} \in [1, N] \setminus \{n\}\ \forall n \in [1, N] \tag{11}$$

$$P(H_I|\mathbf{do}(H_{S_n})) = P(H_I)\ \forall n \in [1, N] \tag{12}$$

H_R, H_I and H_{S_n} $\forall n \in [1, N]$ are the respective representations generated by the router, domain-invariant and N domain-specific modules.

The rules of do-calculus, defined in Pearl (1995), allow one to reduce interventional queries (with the $\mathbf{do}(\cdot)$ operator) to observational queries. We will only use the *deletion of actions* rule. A simplified rule is shown in Equation 13:

$$P(Y|\mathbf{do}(X)) = P(Y) \quad \text{if } (Y \perp\!\!\!\perp X)_{\mathcal{G}_{\overline{X}}} \tag{13}$$

$\mathcal{G}_{\overline{X}}$ represents the causal graph $\mathcal{G}$ with the incoming edges of X removed.

Let us first address the causal relationships of the router. Equations 10 and 9 can be verified using the simplified causal graph in Figure 11. They are a direct application of rule 13. When removing the parents of H_{S_n} or H_I, H_R is **d-separated** (Pearl, 1988) from them: the backward path through C is blocked and the forward path through W_R is not connected to H_{S_n} or H_I. This is due to the use of a separate loss function for training the router when using the vector quantisation routing strategy. One could notice that we do not represent the sum of losses of Equation 4. We omit it in the simplified graph. Its impact on the backward pass is incidental since each element can be optimised independently.

Let us now address the causal relationships of one activated domain-specific module n with its counterparts (Equation 11). Again, under graph $\mathcal{G}_{\overline{H_{S_{\hat{n}}}}}$, H_{S_n}, the backward path through C between is blocked. In addition, we make the assumption that only the module n is activated and is connected

to Y as in Figure 11. This assumption is verified when using the vector quantisation routing strategy. As a consequence, the routing process H_S can be decomposed into multiple subgraphs $H_{S_n} \xrightarrow{H_R} Y$ and $H_{S_{\overline{n}}} \not\xrightarrow{H_R} Y \; \forall \overline{n} \in [1, N] \setminus \{n\}$, with $A \xrightarrow{H_R} B$ equivalent to having $A \rightarrow X \leftarrow H_R$ and $X \rightarrow B$. $A \not\xrightarrow{H_R} B$ removes the second link. Therefore, during the backward pass, there is only one link $\mathcal{L}_Y \rightarrow W_{S_n} \rightarrow H_{S_n}$ and no path to the other domain-specific modules $\overline{n}$. A last type of path can exist; here is an example: assuming a model with two domain-specific modules, S_0 and S_1, activated one after the other, the following path exists: $H_{S_1} \xrightarrow{H_R} Y_\tau \rightarrow \mathcal{L}_{Y_\tau} \rightarrow W_I \xrightarrow{C} H_I \rightarrow Y_{\tau+1} \rightarrow \mathcal{L}_{Y_{\tau+1}} \rightarrow W_{S_2} \xrightarrow{C} H_{S_2}$. There is a causal path forward path from H_{S_1} to H_{S_2}. The path does not exist if there is no invariant module, and Equation 11 holds.

If an invariant module is part of the model, Equations 11 and 12 do not hold because of the path above: H_I and H_{S_n} are not independent in the causal graph $\mathcal{G}_{H_{S_n}}$. Independence is achieved by minimising the Mutual Information between H_I and H_{S_n}, as discussed in the main paper.

C Additional Routing Strategies

In our main experiments, we use a simple routing strategy based on computing vector quantisation from Euclidean distance. In this section, we consider several additional routing strategies:

- K-Means Clustering
- Euclidean Distance Weighting

K-Means Clustering This strategy computes the clusters using K-Means (Lloyd's algorithm) (Lloyd, 1982). We first learn the cluster centroids on the training set separately. Then, we fine-tune the other modules. Therefore, the clustering mechanism is independent of the gradient descent during fine-tuning, and the quality of the clusters with respect to the data distribution depends solely on the robustness of the clustering method. For efficiency reasons, we do not directly perform the clustering on the hidden states of the router module. Before clustering an input embedding, we project it to a more dense space with fewer dimensions (typically 64). We apply Multidimensional Scaling with the SMACOF algorithm (Borg and Groenen, 2005). The algorithm requires us to provide a base of the input space to perform the projection. We span the space using a random set of vectors from the training space. Because the distribution is skewed, we do not have a warranty to build a base. To remain computation-efficient, we sample $8 \times M$ vectors with M the dimensionality of the reduced space.

Euclidean Distance Weighting This strategy differs from the other ones as it does not use vector quantisation. Instead, we compute the Euclidean distance between the embeddings and the centroid coordinates (randomly initialised) and use softmin to convert the distances into continuous weights between zero and one. The lower the distance between the embedding and a centroid, the higher the weight the corresponding domain-specific module will have on the output. Consequently, with this method, all domain-specific modules are always activated. This operation is differentiable and is the closest to the routing process of Mixture-of-Expert models like the Switch Transformer (Fedus et al., 2022b). This method does not follow the causal structure discussed in Section 4. Instead, it uses the output loss to update the centroid coordinates.

The results obtained with these two routing strategies are provided in Appendix E.3.

D Additional Aggregation Schemes

In this section, we describe two additional aggregation schemes between the domain-invariant and domain-specific modules. Instead of using a shared language modelling head, we propose to use a separate head for each module and combine their outputs at the end by a weighted sum. This method tackles the issue of prioritised modules (e.g. one module being overused at the expense of the others). However, the information from the modules is not linearly combined but added separately, reducing expressivity.

We want to bound the output of each model such that it influences the final prediction by a predetermined factor (given by the router output for the domain-specific modules and provided as a hyperparameter for the domain-invariant module). Each module outputs unbounded logits. The lack of bounds

prevents them from directly multiplying the logits by their weighting factor and summing them together. Indeed, one module could overcome the weighting by increasing the magnitude of its logits. We consider two combination schemes: in the *logit space* and in the *probability space*.

Combination in the logit space The aggregation scheme in the logit space is very similar to the one performed in the latent space in the main paper. We first perform a shared batch normalisation (Ioffe and Szegedy, 2015) between the modules to overcome the unbounded issue in the logit space. For a batch of size $|\mathcal{B}|$, one domain-specific active module and one domain-invariant module, batch normalisation is operated on $2 \times |\mathcal{B}|$ samples. We attribute a weight w_I to the domain-invariant module as a hyperparameter and $w_{S_n} = r_n \cdot (1 - w_I)$ to the domain-specific module, with r_n the weight given by the router. After normalisation, We multiply each logit value by its corresponding weight and sum them together.

Combination in the probability space Each module outputs unbounded logits, so we first convert each output into normalised probabilities (that sum to one). We then perform the weighting in each probability space before converting them back to logits (shown in Equation 14). Finally, the outputs from all modules are summed together (shown in Equation 15). The final probabilities are shown in Equation 16.

$$\widetilde{l_a}(Y|c) = \log\left(\frac{1 - w_a}{2} + w_a \cdot P_a(Y|c)\right) + \log(B_a) \tag{14}$$

$$l(Y|c) = \widetilde{l_I}(Y|c) + \sum_{n \in [1,N]} \widetilde{l_{S_n}}(Y|c) \tag{15}$$

$$P(Y|c) = \sigma(l(Y|c)) \tag{16}$$

c is the input context. $P(Y|c)$ is the final output distribution between all words Y, obtained using softmax normalisation σ on the output logits $l(Y|c)$. The output logits are obtained by summing the weighted logits of the domain-invariant module $\widetilde{l_I}(Y|c)$ and all domain-specific modules $\widetilde{l_{S_n}}(Y|c)$. Equation 14 shows the weighting process for all modules ($a \in \{I, S_1, \ldots, S_N\}$). The weight w_I is a hyperparameter set prior to training. The weights $w_{S_n} \forall\, n \in [1, N]$ combine the weight w_I with the router weights r_n: $w_{S_n} = r_n \cdot (1 - w_I)$. B_a is a normalisation term that ensures the conversion function between probabilities and logits is invertible.

The results obtained with these two aggregation schemes are provided in Appendix E.4.

E Additional Experiments

E.1 Evolution of the Mutual Information Across Training

To ensure the independence between the domain-specific and domain-invariant modules, we minimise the mutual Information between them. Figure 12 shows the evolution of Mutual Information during training. We observe that it quickly decreases to reach below $0,0001$. Figure 13 shows the same loss for the variants using aggregation in the logit and probability spaces. Unlike for the main model, we observe small spikes in the loss after 100 training steps. The aggregation scheme that uses a shared language modelling head (our default) seems more stable during training.

E.2 Routing Alignment and Visualisation

We study the module attribution performed by the router more deeply. Table 3 shows the alignment between the two domain-specific modules. We first observe that the division is mainly syntactic: each module specialises towards one type of input format, either text or symbolic. It aligns perfectly with the dataset.

Figures 14, 15, 16 and 17 show visualisations of the clusters in a 2D space. Figures 14 and 15 show the i.i.d and o.o.d sets of ACRE. Figures 16 and 17 show the i.i.d and o.o.d sets of RAVEN. As in the main paper, the projection is made using Multidimensional Scaling (MDS) (Borg and Groenen, 2005). For illustration purposes, we observe the clusters formed by the K-Means method for $N = 4$ modules. We also observe the clusters formed from the penultimate hidden states of the router. As discussed above

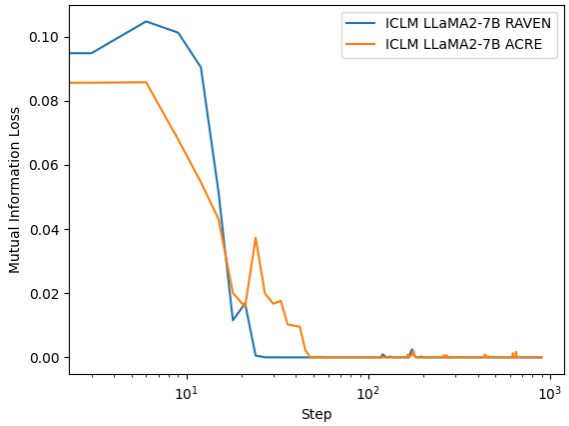


Figure 12: Evolution of the Mutual Information loss during training on ACRE and RAVEN. The x-axis corresponds to the number of training steps and is shown in the log scale.

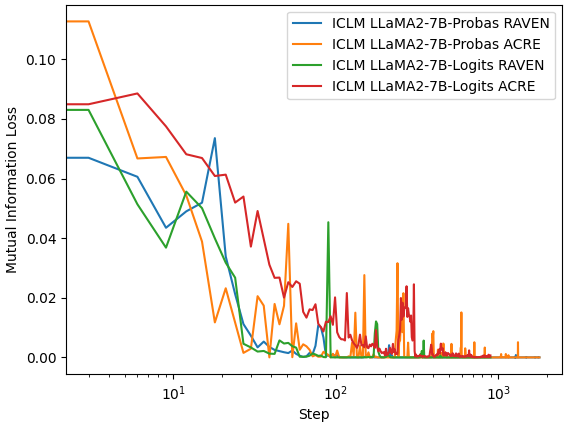


Figure 13: Evolution of the Mutual Information loss during training of variants with probability and logits aggregation on ACRE and RAVEN. The x-axis corresponds to the number of training steps and is shown in the log scale.

Table 3: Alignment between modules and formats in the ACRE dataset. Each column shows the proportion of activation for each module for a given dataset. Each module specialises perfectly to one dataset.

	ACRE		-Comp		-Sys	
	Text	Symb	Text	Symb	Text	Symb
$n = 0$	0.0	1.0	0.0	1.0	0.0	1.0
$n = 1$	1.0	0.0	1.0	0.0	1.0	0.0

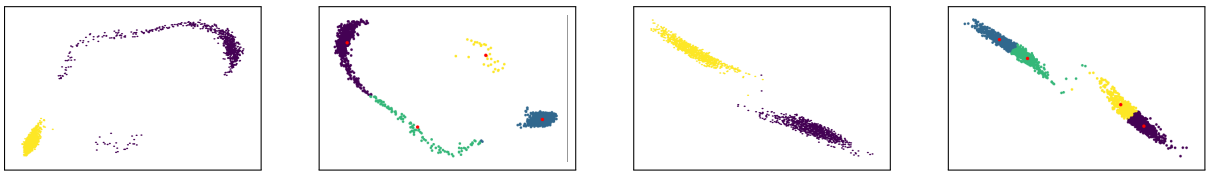
and in the main paper, there is a clear division between text and symbolic embeddings, but the o.o.d sets are not well separated in ACRE while they are in RAVEN. This division (and absence of division) is also present in the previous hidden states, although the separation is less obvious: all embeddings tend to align to a single axis.

We want to study the router's behaviour further when faced with a diverse set of input data. To this end, we feed six different datasets to the model: the i.i.d text and symbolic sets of ACRE and RAVEN, PVR (Zhang et al., 2021b) and ARC (Chollet, 2019) datasets. The visualisations are in Figure 18. Overall, the datasets are well separated but have different shapes. While some form dense amalgamates, others spread in the latent space. The observations from ACRE and RAVEN suggest that the distance in the embedding space between a module cluster and an input can be an indicator of the module's performance on the input. The o.o.d sets of ACRE are merged in the latent space, and the model maintains accuracy across the sets. In parallel, the o.o.d sets of RAVEN are separated by clear boundaries, and the accuracy drops as the distance with the i.i.d set increases. Experiments on a larger scale are needed to validate or invalidate the hypothesis and discriminate the true causes responsible for this behaviour from spurious correlations.

E.3 Variations of the Routing Strategy

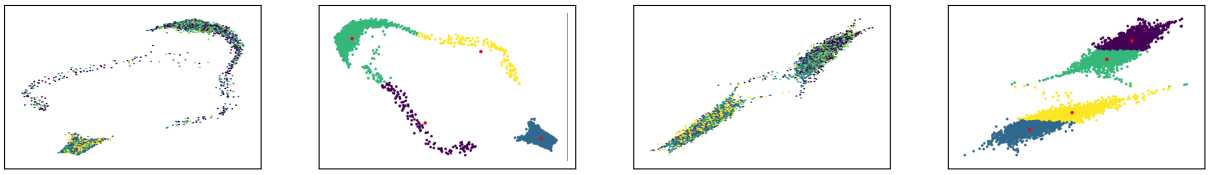
We perform additional experiments on ACRE and RAVEN datasets using the routing strategies introduced in Appendix C: K-Means and weighting. Tables 4 and 5 show the results.

The alternative routing strategies achieve similar and sometimes superior performance than the base ICLM model. As observed in the previous section, the router creates well-defined clusters that the K-Means and Euclidean distance vector quantisation strategies tend to follow. No explicit differentiation of the routing process can be observed from the visualisations. The difference in performance may lie in the optimisation process. K-Means does not backpropagate information to the router; weighting backpropagates from the output loss, and vector quantisation backpropagates from a secondary loss.



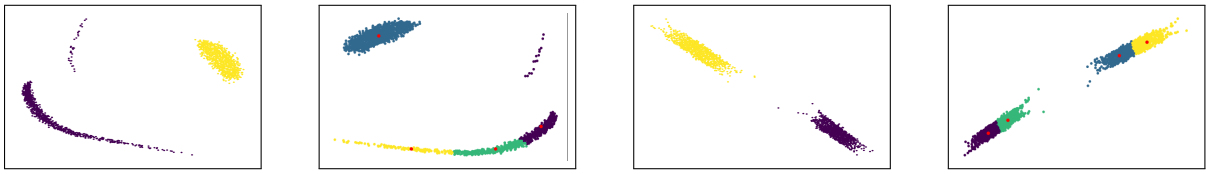
(a) Ground Truth ($l = -1$) (b) K-Means ($l = -1$) (c) Ground Truth ($l = -2$) (d) K-Means ($l = -2$)

Figure 14: Clusters formed from the hidden states of LLaMA2 on the ACRE training sets. The visualisations contain the last two levels (l) of hidden layers. The ground truth shows the true splits (text/symbolic). The learned clusters use 4 centroids.



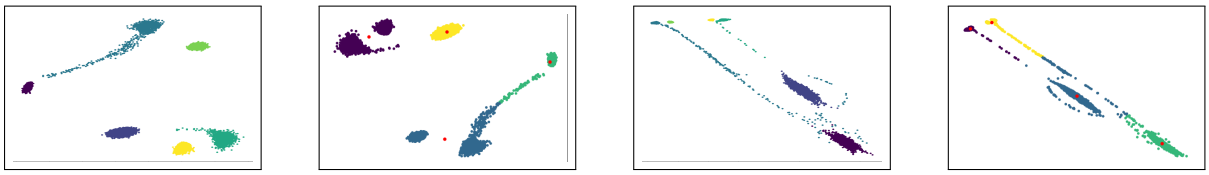
(a) Ground Truth ($l = -1$) (b) K-Means ($l = -1$) (c) Ground Truth ($l = -2$) (d) K-Means ($l = -2$)

Figure 15: Clusters formed from the hidden states of LLaMA2 on the ACRE o.o.d sets. The visualisations contain the last two levels (l) of hidden layers. The ground truth shows the true splits (text/symbolic/o.o.d splits). The learned clusters use 4 centroids.



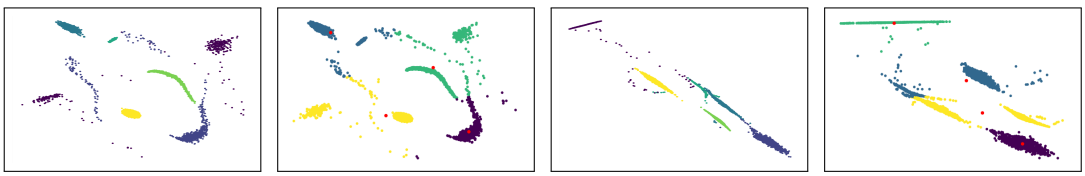
(a) Ground Truth ($l = -1$) (b) K-Means ($l = -1$) (c) Ground Truth ($l = -2$) (d) K-Means ($l = -2$)

Figure 16: Clusters formed from the hidden states of LLaMA2 on the RAVEN training sets. The visualisations contain the last two levels (l) of hidden layers. The ground truth shows the true splits (text/symbolic). The learned clusters use 4 centroids.



(a) Ground Truth ($l = -1$) (b) K-Means ($l = -1$) (c) Ground Truth ($l = -2$) (d) K-Means ($l = -2$)

Figure 17: Clusters formed from the hidden states of LLaMA2 on the RAVEN o.o.d sets. The visualisations contain the last two levels (l) of hidden layers. The ground truth shows the true splits (text/symbolic). The learned clusters use 4 centroids.



(a) Ground Truth ($l = -1$) (b) K-Means ($l = -1$) (c) Ground Truth ($l = -2$) (d) K-Means ($l = -2$)

Figure 18: Clusters formed from the hidden states of LLaMA2 on the training sets of ACRE, ARC, PVR and RAVEN. The visualisations contain the last two levels (l) of hidden layers. The ground truth shows the true splits between each dataset. The learned clusters use 4 centroids.

Table 4: Accuracy on the ACRE i.i.d and o.o.d test sets. Datasets are represented in columns, and models in rows. ICLM is trained on text and symbolic i.i.d training sets. Models with a * indicate that the results are introduced in this paper. The best model is shown in **bold**.

	ACRE		-o.o.d-Comp		-o.o.d-Sys	
	Text	Symb	Text	Symb	Text	Symb
ICLM* (ours)	0.653	**0.950**	0.663	0.931	0.634	0.901
ICLM-Weighted* (ours)	0.812	0.921	0.802	**0.960**	0.842	**0.970**
ICLM-K-Means* (ours)	**0.901**	0.881	**0.911**	0.911	**0.891**	0.921

Table 5: Accuracy on the RAVEN i.i.d and o.o.d test sets. The characteristics are the same as in Table 4.

	RAVEN		-o.o.d-Four		-o.o.d-In-Center	
	Text	Symb	Text	Symb	Text	Symb
ICLM* (ours)	**1.000**	0.980	0.703	**0.703**	0.515	0.228
ICLM-Weighted* (ours)	**1.000**	**1.000**	**0.743**	**0.703**	**0.653**	0.248
ICLM-K-Means* (ours)	**1.000**	**1.000**	0.634	0.673	0.515	**0.287**

E.4 Variations of the Aggregation Scheme

We perform additional experiments on ACRE and RAVEN datasets using the aggregation schemes introduced in Appendix D: in the logit and probability spaces. Tables 4 and 5 show the results.

Table 6: Accuracy on the ACRE i.i.d and o.o.d test sets.

	ACRE		-o.o.d-Comp		-o.o.d-Sys	
	Text	Symb	Text	Symb	Text	Symb
ICLM* (ours)	0.653	**0.950**	0.663	**0.931**	0.634	0.901
ICLM-Logits* (ours)	0.842	**0.950**	**0.881**	0.901	**0.901**	0.921
ICLM-Probas* (ours)	**0.921**	**0.950**	0.832	0.921	0.891	**0.931**

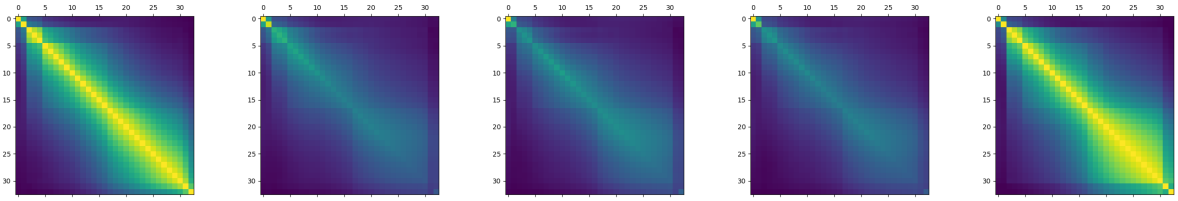
Table 7: Accuracy on the RAVEN i.i.d and o.o.d test sets.

	RAVEN		-o.o.d-Four		-o.o.d-In-Center	
	Text	Symb	Text	Symb	Text	Symb
ICLM* (ours)	**1.000**	0.980	**0.703**	0.703	0.515	0.228
ICLM-Logits* (ours)	**1.000**	**0.990**	0.644	**0.713**	**0.703**	0.268
ICLM-Probas* (ours)	0.802	0.931	0.614	0.624	0.495	**0.297**

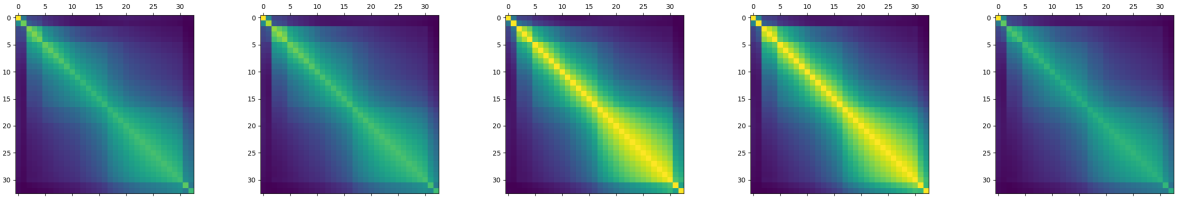
As per the routing strategies, the alternative aggregation schemes achieve similar and sometimes superior performance than the base ICLM model. No scheme is systematically better than the others. These results show that less expressive aggregation methods, i.e. weighted sums, can perform similarly to trained dense layers on abstract and causal reasoning tasks.

E.5 Module Correlation Across Hidden States

To further investigate the level of Independence of the LLM modules, we measure the Pearson Correlation Coefficient between all hidden states of the domain-invariant and domain-specific modules during inference on ACRE and RAVEN. Figures 19 and 20 show the results. The intra-module correlations (Figures 19a, 19e, 19h, 19i, 20a, 20e, 20h and 20i) show that hidden states from close layers in the model are highly correlated. Furthermore, correlation *blocks* are visible, i.e. sequences of layers that demonstrate a high level of correlation between them and a low level of correlation with the layers not in the sequence. We observe five correlation blocks well-defined on ACRE and with fuzzy-edges on RAVEN.



(a) Router-Router. (b) Router-Inv. (c) Router-Dom 0. (d) Router-Dom 1. (e) Inv-Inv.

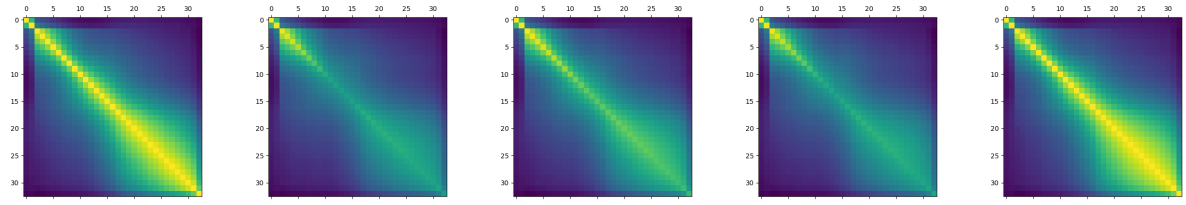


(f) Inv-Dom 0. (g) Inv-Dom 1. (h) Dom 0-Dom 0. (i) Dom 1-Dom 1. (j) Dom 0-Dom 1.

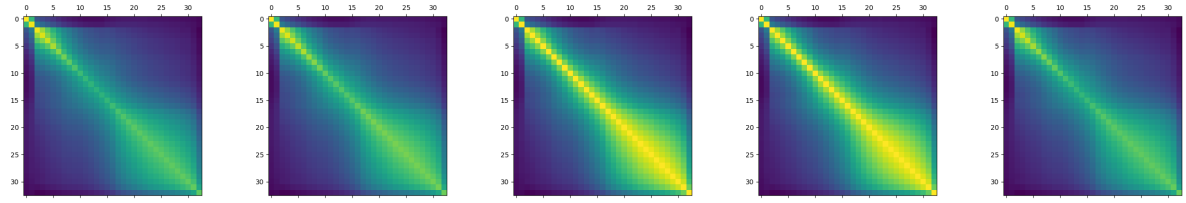
Figure 19: Measures of the Pearson Correlation Coefficient between module hidden states during inference on ACRE dataset. Rows and columns represent layers 0 to 33 of a LLaMA2 module. Router refers to the routing module, Inv refers to the domain-invariant module, and Dom *i* refers to the *i* domain-specific module. The caption indicates the modules used for each row-column pair.

These correlation blocks hold in the inter-module correlation matrices, indicating that similar mechanisms are shared across modules. However, the fine-tuning and regularisation procedures reduce the influence of the shared mechanisms and enhance specialisation.

Correlation Measures in Transfer Learning Settings We perform the same measures in the transfer learning settings. Figures 21 and 22 show the results. The studied ICLM model has four modules: while domain-2 and dmain-3 are well aligned with the respective text and symbolic RAVEN splits, the two remaining modules do not fully align with the two ACRE splits. Particularly, the domain-2 shows a very low level of correlation with the other modules and even with its own layers. This visualisation of the layers confirms the results observed in Table 2 and demonstrates that the poor performance of this module is due to a collapse of the module during the transfer learning: the module inputs are no longer correlated with the module outputs.

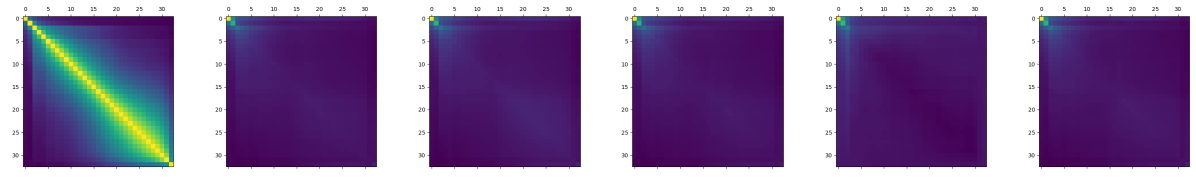


(a) Router-Router. (b) Router-Inv. (c) Router-Dom 0. (d) Router-Dom 1. (e) Inv-Inv.

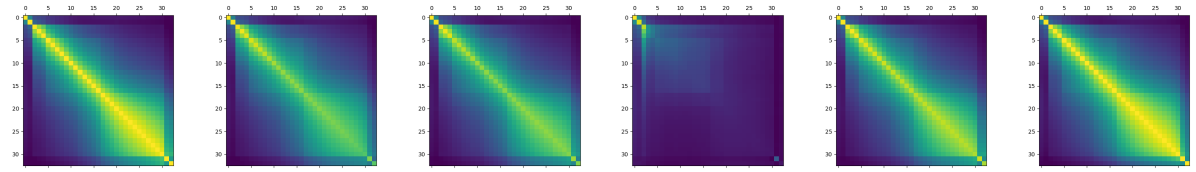


(f) Inv-Dom 0. (g) Inv-Dom 1. (h) Dom 0-Dom 0. (i) Dom 1-Dom 1. (j) Dom 0-Dom 1.

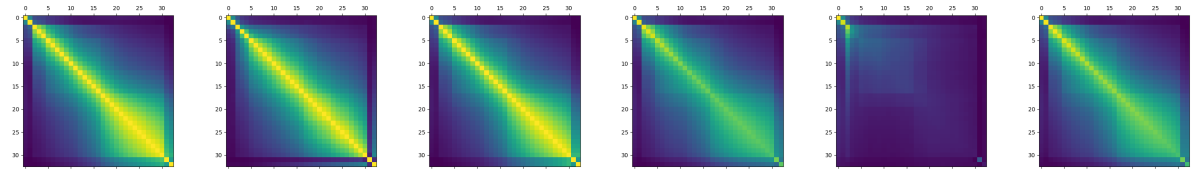
Figure 20: Measures of the Pearson Correlation Coefficient between module hidden states during inference on RAVEN dataset. Rows and columns represent layers 0 to 33 of a LLaMA2 module. Router refers to the routing module, Inv refers to the domain-invariant module, and Dom *i* refers to the domain-specific module *i*. The caption indicates the modules used for each row-column pair.



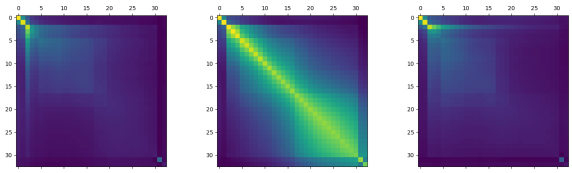
(a) Router-Router. (b) Router-Inv. (c) Router-Dom 0. (d) Router-Dom 1. (e) Router-Dom 2. (f) Router-Dom 3.



(g) Inv-Inv. (h) Inv-Dom 0. (i) Inv-Dom 1. (j) Inv-Dom 2. (k) Inv-Dom 3. (l) Dom 0-Dom 0.

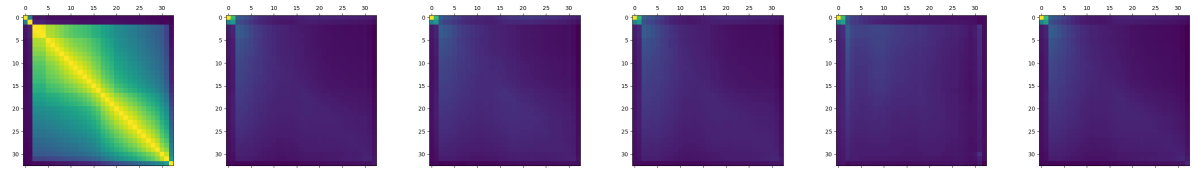


(m) Dom 1-Dom 1. (n) Dom 2-Dom 2. (o) Dom 3-Dom 3. (p) Dom 0-Dom 1. (q) Dom 0-Dom 2. (r) Dom 0-Dom 3.

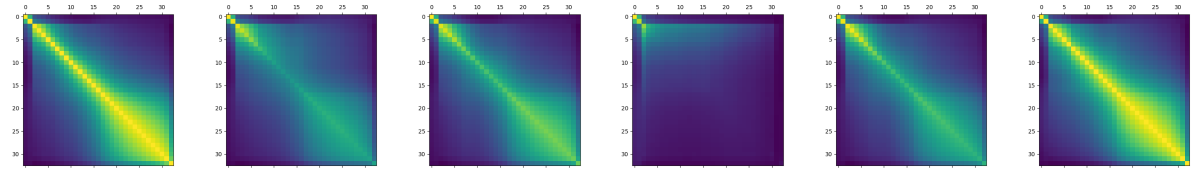


(s) Dom 1-Dom 2. (t) Dom 1-Dom 3. (u) Dom 2-Dom 3.

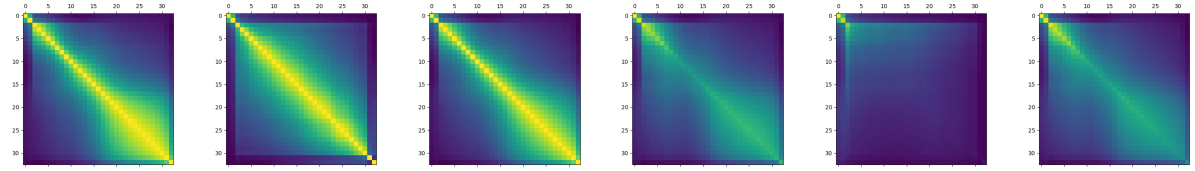
Figure 21: Measures of the Pearson Correlation Coefficient between module hidden states of the transfer learning model during inference on ACRE dataset. Rows and columns represent layers 0 to 33 of a LLaMA2 module. Router refers to the routing module, Inv refers to the domain-invariant module, and Dom *i* refers to the *i* domain-specific module. The caption indicates the modules used for each row-column pair.



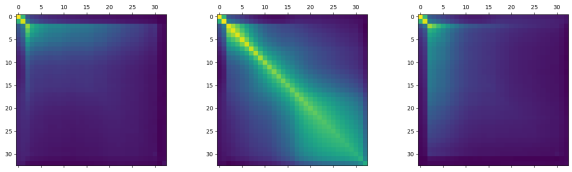
(a) Router-Router. (b) Router-Inv. (c) Router-Dom 0. (d) Router-Dom 1. (e) Router-Dom 2. (f) Router-Dom 3.



(g) Inv-Inv. (h) Inv-Dom 0. (i) Inv-Dom 1. (j) Inv-Dom 2. (k) Inv-Dom 3. (l) Dom 0-Dom 0.



(m) Dom 1-Dom 1. (n) Dom 2-Dom 2. (o) Dom 3-Dom 3. (p) Dom 0-Dom 1. (q) Dom 0-Dom 2. (r) Dom 0-Dom 3.



(s) Dom 1-Dom 2. (t) Dom 1-Dom 3. (u) Dom 2-Dom 3.

Figure 22: Measures of the Pearson Correlation Coefficient between module hidden states of the transfer learning model during inference on RAVEN dataset. Rows and columns represent layers 0 to 33 of a LLaMA2 module. Router refers to the routing module, Inv refers to the domain-invariant module, and Dom *i* refers to the *i* domain-specific module. The caption indicates the modules used for each row-column pair.